Assessing Similarity Value between Two Ontologies

Aly Ngoné Ngom[1], Guidedi Kaladzavi[1,2], Fatou Kamara-Sangaré[1] and Moussa Lo[1]
[1]*LANI, Gaston Berger University, BP 234, Saint - Louis, Senegal*
[2]*University of Maroua, Cameroon*

Keywords: Ontologies Similarity, Concepts Similarity, Semantic Similarity.

Abstract: The aim of this paper is to present an appraoch for assessing similarity between ontolgies. This approach is based on set theory, edges-based semantic similarity and features based similarity. We first determine the set of concepts that is shared by two ontologies and the sets of concepts that are different from them. Then we extend ontologies by using the set of concepts shared by the two ontologies. Then we redetermine set of concepts shared by the two extended ontologies. finally, we end with the assessment of the similarity between ontologies by using the average values of similarity of the sets of specific concepts to each not extended ontologies, and the set of concepts shared by extended ontologies.

1 INTRODUCTION

Ontologies allow to formalize knowledge related to the description of the world by making them accessible and shareable across the Web. They introduce the semantic layer into the architecture of the on based-systems (Dramé, 2014). When several ontologies are used for an application, it is necessary that these ontologies present some similarity. The assessing of similarity between ontologies may be very interesting. Indeed, it can make easy the choice of ontologies in the case of elaboration of a system, which uses them. In addition, it can help to evaluate the ontology evolution by comparing its different versions. In (Ngom et al., 2017), we have proposed an approach for assessing similarity between two ontolgies O_1 and O_2. This approach has given good results. However, it would be interesting to take into account some properties (relations, axioms) for extending the formed sets (set of resemblance and set of differences) in the goal to improve this approach. The aim of this paper is to present another approach which improves our proposition in (Ngom et al., 2017).
This paper continues by presenting our research context. Section 2 presents the related work. Then, we present our approach in section 3 before making its experimentation in section 4. In section 5, we analyse the results obtained by the experimentation and compare them to the results obtained in (Ngom et al., 2017). We end with conclusion and perspectives of this work.

2 RELATED WORKS

There are several works, which are dedicated to the evaluation of similarity between two concepts in an ontology. However, there are not many works which deal with evaluation of similarity between ontologies. The following are some works about similarity between two ontologies.
Maedche and Staab (Maedche and Staab, 2002) propose a method for comparing two ontologies. This method is based on two levels:

- the **Lexical level** which consists of investigation on how terms are used to convey meanings ;
- the **Conceptual level** which is the investigation of what conceptual relations exist between terms.

The Lexical comparison allows to find concepts by assessing syntaxic similarity between concepts. It is based on Levenshtein (Levenshtein., 1966) edit distance (ed) formula which allows to measure the minimum number of change required to transform one string into another, by using a dynamic programming algorithm. The Conceptual Comparison Level allows to compare the semantic of structures of two ontologies. Authors use Upwards Cotopy (UC) to compare the Concept Match (CM). Then, they use the CM to determine the Relation Overlap (RO). Finally they assess the average of RO.

This approach allows to assess similarity between two ontologies by using the Lexical and Conceptual Comparison Level. However, if we reverse the posi-

Ngom, A., Kaladzavi, G., Kamara-Sangaré, F. and Lo, M.
Assessing Similarity Value between Two Ontologies.
DOI: 10.5220/0006959903430350
In *Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2018) - Volume 2: KEOD*, pages 343-350
ISBN: 978-989-758-330-8

tion of some concepts in the hierarchy, we can get the same results because the method only considers the presence of the concept in the hierarchy.

In (Ngom et al., 2017), we have proposed an approch which assesses similaty between two ontologies. The approach is based on set theory, edges based (Ngom., 2015) and feature-based similarity based (Tversky, 1977). It is composed of three steps. The first step consists to determine the different sets : set of concepts shared by the two ontologies and sets of concepts specific to each ontology. In the second step, we have assessed the average similarity values between concepts for each sets thanks to semantic similarity measures. Finally, in the last step, we have assessed similarity between ontologies by redefining Tversky's measure relying to the two first steps.

To assess similarity between two ontologies, we have defined a measure which readjust the Tversky measure. This measure takes into acount shared features and differences of ontologies. Applying the Tversky measure, the similarity between O_1 and O_2 is given by the formula 1.

$$Tvr_{(O_1,O_2)} = \frac{f(O_1 \cap O_2)}{f(O_1 \cap O_2) + \alpha . f(O_1 \backslash O_2) + \beta . f(O_2 \backslash O_1)} \quad (1)$$

Instead of the function f, we use Wu and Palmer (Wu and Palmer, 1994) semantic similarity measure. for every determined set, we have computed the average of the similarity values between concepts. Using Wu and Palmer similarity measure, the similarity between two concepts c_1 and c_2 is given by the formula 2.

$$Sim(c_1,c_2) = \frac{2 \times depth(c_3)}{depth(c_1) + depth(c_2)} \quad (2)$$

The concept c_3 represents the Least Common Subsumer (LCS) of concepts c_1 and c_2. By replacing the terms of the Tversky measure with the average of the similarity values between concepts of the determined sets, formula 1 becomes formula 3.

$$T_{Ngom}(O_1,O_2) = \frac{\theta . \bar{x}_{O_1} + \omega . \bar{x}_{O_2}}{\theta . \bar{x}_{O_1} + \omega . \bar{x}_{O_2} + \alpha . \bar{y}_{(O_1 \backslash O_2)} + \beta . \bar{z}_{(O_2 \backslash O_1)}} \quad (3)$$

with :

- $\theta = \frac{cardinality(O_1 \cap O_2)}{cardinality(O_1)}$;
- $\omega = \frac{cardinality(O_1 \cap O_2)}{cardinality(O_2)}$;
- $\alpha = \frac{cardinality(O_1 \backslash O_2)}{cardinality(O_1)}$;
- and $\beta = \frac{cardinality(O_2 \backslash O_1)}{cardinality(O_2)}$;
- $cardinality(O)$ is the number of elements (concepts) of the set (ontology) O ;

and where :

- $\bar{x}_{O_1}$ (respectively $\bar{x}_{O_2}$) is the average value of similarity between concepts (x_i,x_j) in ontology O_1 (respectively (x_i,x_j) in ontology O_2). $i,j \in \mathbb{N}$ and $i \neq j$.
- $\bar{y}_{(O_1 \backslash O_2)}$ (respectively $\bar{z}_{(O_2 \backslash O_1)}$) is the average value of similarity between concepts (y_i,y_j) (respectively (z_i,z_j)) present in ontology O_1 but not in O_2 (respectively present in ontology O_2 but not in O_1). $i,j \in \mathbb{N}$ and $i \neq j$.
- the coefficients θ, ω, α and β allow to take into account the similarity values in relation to the number of concepts of the sets and number of concepts of ontologies.

The measure presented by formula 3 respects this properties :

- the measure is symetric : $T_{Ngom}(O_1,O_2) = T_{Ngom}(O_2,O_1)$;
- the measure is bounded between 0 and 1 ;
- if $T_{Ngom}(O_1,O_2) = 1$ then $O_1 = O_2$.

The method we have proposed in (Ngom et al., 2017) gives satisfactory results. Indeed, it allows to assess similarity between two ontologies while taking into account the semantic links that exist between the concepts in ontologies. However, it doesn't take into account properties of concepts for extending the formed sets (set of resemblance and set of difference). In this paper, we propose another approach which takes into account relation "is-a" between concepts for the extension of the set of concepts shared by O_1 and O_2.

3 OUR APPROACH

3.1 Principe

The approach we propose is an improvement of our previous paper (Ngom et al., 2017). As in our previous works, this approach is based on set theory, edges based (Ngom., 2015) and feature-based similarity based (Tversky, 1977). It can be summarized in five steps :

- **Step 1** consists to determine the sets $(O_1 \backslash O_2)$, $(O_2 \backslash O_1)$ and $(O_1 \wedge O_2)$.
- Once the sets are determined, we assess the average of the semantic similarity values between concepts of each set in **step 2**.
- In **step 3**, we extend ontologies O_1 and O_2 by using the set $(O_1 \wedge O_2)$. In this step, for each concept c of $(O_1 \wedge O_2)$, we search there sons x_i $(i \in \mathbb{N})$ in O_1 (respetively O_2) and we add them

as sons of c in O_2 (respetively O_1) if they don't exist in this ontology. At the end of this step, we obtain two ontologies : O'_1 (respectively O'_2) which extends O_1 (respectively O_2) with concepts of O_2 (respectively O_1). Thus, extension of ontologies allows us to determine the set of concepts $(O'_1 \wedge O'_2)$ shared by the two ontologies.

- In **Step 4**, we determine $(O'_1 \wedge O'_2)$ which is the set of shared concepts by ontologies O'_1 and O'_2.
- Finally, in the **step 5**, we assess similarity between ontologies by using the results of the step 2 and 4 in our measure which is a redefinition of the T_{Ngom} measure (Ngom et al., 2017).

In summary, for assessing similarity between ontologies, we use sets $(O_1 \backslash O_2)$, $(O_2 \backslash O_1)$ and $(O'_1 \wedge O'_2)$; i.e we consider the difference between O_1 and O_2 by using sets $(O_1 \backslash O_2)$ and $(O_2 \backslash O_1)$, and the resemblance between the two ontologies by using set $(O'_1 \wedge O'_2)$. Figure 1 represents the differents that we use for assessing similarity beetween ontologies O_1 and O_2. In

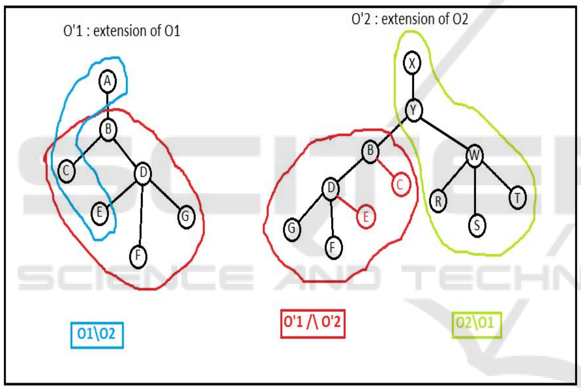

Figure 1: Representation of extensions of ontologies O'_1 and O'_2 with Tversky's feature model.

figure 1, we distinguish three parts :

- $(O_1 \backslash O_2) = \{A,C,E\}$: set of concepts present in O_1 and not in O_2 ;
- $(O_2 \backslash O_1) = \{R,S,T,W,X,Y\}$: set of concepts present in O_2 and not in O_1 ;
- $(O'_1 \wedge O'_2) = \{B,C,D,E,F,G\}$: set of concepts present in O'_1 and O'_2.

3.2 Measure

The measure we present in this paper is an improvement of our measure T_{Ngom} (Ngom et al., 2017) which redifines Tversky's (Tversky, 1977) similarity measure. For assessing similarity between two ontologies O_1 and O_2, our measure takes into account the difference between the two ontologies by assessing the average similarity values of sets $(O_1 \backslash O_2)$ and $(O_2 \backslash O_1)$, and their common concepts by extending the ontologies (O_1 to O'_1 and O_2 to O'_2) and assessing the average similarity values of the set of common concepts of extended ontologies $((O'_1 \wedge O'_2))$. We use Wu and Palmer measure (Wu and Palmer, 1994) for computing semantic similarity between concepts of sets in ontologies. The measure is given by the formula 4:

$$N_{Plus}(O_1,O_2) = \frac{(\theta.\bar{x}_{O'_1}+I_2)+(\omega.\bar{x}_{O'_2}+I_1)}{(\theta.\bar{x}_{O'_1}+I_2)+(\omega.\bar{x}_{O'_2}+I_1)+\alpha.\bar{y}_{(O_1\backslash O_2)}+\beta.\bar{z}_{(O_2\backslash O_1)}} \quad (4)$$

with :

- $\theta = \dfrac{cardinality(O'_1 \cap O'_2)}{cardinality(O_1)+n_1+n_2}$;
- $\omega = \dfrac{cardinality(O'_1 \cap O'_2)}{cardinality(O_2)+n_1+n_2}$;
- $\alpha = \dfrac{cardinality(O_1 \backslash O_2)}{cardinality(O_1)}$;
- $\beta = \dfrac{cardinality(O_2 \backslash O_1)}{cardinality(O_2)}$;
- $I_1 = \dfrac{1}{1+n_2}$;
- $I_2 = \dfrac{1}{1+n_1}$;
- $\bar{x}_{O'_1}$ (respectively $\bar{x}_{O'_2}$) is the average value of similarity between concepts (x_i,x_j) in ontology O'_1 (respectively (x_i,x_j) in ontology O'_2). $i,j \in \mathbb{N}$ and $i \neq j$.
- $\bar{y}_{(O_1\backslash O_2)}$ (respectively $\bar{z}_{(O_2\backslash O_1)}$) is the average value of similarity between concepts (y_i,y_j) (respectively (z_i,z_j)) present in ontology O_1 but not in O_2 (respectively present in ontology O_2 but not in O_1). $i,j \in \mathbb{N}$ and $i \neq j$.
- $cardinality(O)$ is the number of elements (concepts) of the set (ontology) O ;
- I_i : Integrity coefficient of Ontology O_i $(i \in \mathbb{N})$;
- n_i : number of concepts of O_i added for extending O_j $(i,j \in \mathbb{N})$;
- As in (Ngom et al., 2017), θ, ω, α and β are parameters which allow to take into account the similarity values in relation to the number of concepts of the sets and number of concepts of ontologies.

The integrity coefficient of Ontology (I_i) is a value which is related to the number of concepts of ontology

O_j (n_j) that we have to add to O_i in goal to extend it ($i,j \in \mathbb{N}$). The larger is n_j, the smaller is I_i. We have the expression 5:

$$\begin{cases} \lim_{n\to\infty} I = \lim_{n\to\infty} \frac{1}{1+n} = 0; \\ \lim_{n\to 0} I = \lim_{n\to 0} \frac{1}{1+n} = 1; \end{cases} \quad (5)$$

with ($n \in \mathbb{N}$).

We note that measure presented by formula 4 like formula 3 respects this properties :

- the measure is symetric : $N_{Plus}(O_1,O_2) = N_{Plus}(O_2,O_1)$;
- the measure is bounded between 0 and 1 ;
- if $N_{Plus}(O_1,O_2) = 1$ then $O_1 = O_2$.

In the next section, we will introduce some important algorithms.

3.3 Algorithms

In this section, we will present the important algorithms that we can implement to assess similarity between ontologies. The algorithms we define here are based on different steps that we have mentioned in section 3.1 to evaluate the similarity between the ontologies. The algorithms 1 and 2 allow respectively to form the sets of concepts that represent differences and resemblances between two ontologies O_1 and O_2. Algorithm 1 implements a method named $diffOnto(stackO_1, stackO_2)$ which allows to extract the difference between two ontologies. In input, we have two set of concepts stored on stacks. $stackO_1$ (respectively $stackO_2$) store all O_1's concepts (respectively O_2's concepts). The function $checkConcept(c, stackO_2)$ checks if concepts c is present in ontology O_2. If c is not in O_2, then c will be added in the stack of difference $stackDiff$. In output, the method $diffOnto(stackO_1, stackO_2)$ returns a stack of concepts which represents all concepts present in O_1 and not in O_2.

Algorithm 1: diffOnto. /* Differences between two ontologies */

```
1    input  : $stackO_1$, $stackO_2$ : stack of concpts;
     output: $stackDiff$ : stack of concepts;
2  variables : $c$ : concept ; $stackDiff$ : stack of
     concepts;
3  while (!$stackO_1$) do
4      $c$ ←depilate (stackO$_1$) ;
5      if (!$checkConcept(c, stackO_2)$) then
6          $stack(c, stackDiff)$ ;
7      end
8  end
9  return $stackDiff$ ;
```

Algorithm 2: cCOnto. /* (Common Concept of Ontologies) Resemblance between two ontologies */

```
     input  : $stackO_1$, $stackO_2$ : stack of concpts;
     output: $stackCommon$ : stack of concepts;
1  variables : $c$ : concept ; $stackDiff$ : stack of
     concepts;
2  if ($sizeOf(stackO_1) \leqslant sizeOf(stackO_2)$) then
3      while (!$stackO_1$) do
4          $c$ ←depilate (stackO$_1$) ;
5          if ($checkConcept(c, stackO_2)$) then
6              $stack(c, stackCommon)$ ;
7          end
8      end
9  end
10 else
11     while (!$stackO_2$) do
12         $c$ ←depilate (stackO$_2$) ;
13         if ($checkConcept(c, stackO_1)$) then
14             $stack(c, stackCommon)$ ;
15         end
16     end
17 end
18 return $stackCommon$ ;
```

Algorithm 2 called $cCOnto(stackO_1, stackO_2)$ allows to get the set of concepts that belong to ontologies O_1 and O_2. In input, as for algorithm 1, we have the stacks of concepts $stackO_1$ and $stackO_2$ which store, respectively, the concepts of O_1 and O_2. The method $sizeOf(stackO_1)$ (respectively $sizeOf(stackO_2)$) gives the size of the stack $stackO_1$ (respectively $stackO_2$) and the method $checkConcept$ checks, if a concept, belongs to the stack of concept. If the result is true, then, the concept is added in the stack $stackCommon$. In output, we have a stack of concepts $stackCommon$ which store all concepts that belong to O_1 and O_2.

Algorithm 3 is defined to assess the average of similarity values between concepts of a set of concepts (set of differences and set of resemblance). In input, we have a stack of concept ($stackConcept$) and an ontology (O). The stack $stackConcept$ represents a set of concepts (set of difference or set of resemblance). The algorithm compute similarity between all concepts of the set and assess the average of values of similarity. For that, we extract the first concept out of the stack and fix a pointer to the new first concept of the stack in the goal to assess similarity between concepts. The function $Sim(c_i, c_j, O)$ ($i, j \in \mathbb{N}$ and $i \neq j$) implements an edge-based semantic similarity measure among measures studied in (Ngom et al., 2017).

```
Algorithm 3: aSV. /* (Average Similarity Value) Assess the average of similarity values between concepts of a set of concepts */
   input  : $stackConcept$ : stack of concpts; $O$ : Ontology ;
   output: $average$ : real;
1  variables : $c_1, c_2$ : concept ;
2  /* $indexStack$ is used as pointer of stack */
3  $indexStack$ : stack of concepts;
4  $meter$ : integer ;
5  $valueSim$ : real ;

6  $meter \leftarrow 0$;
7  $valueSim \leftarrow 0$;
8  while ($!stackConcept$) do
9      $c_1 \leftarrow$depilate (stackConcept);
10     /* to fix the pointer in the first element of the stack of concepts after calling the depilate function, the first element of the stack has changed and becomes the element after $c_1$.*/
11     $indexStack \leftarrow$stackConcept;
12     while ($!indexStack$) do
13         /* To recover the first element of the stack without to depilate it */
14         $c_2 \leftarrow$indexStack$\rightarrow$value;
15         $valueSim \leftarrow$valueSim+Sim($c_1, c_2, O$) ;
16         $meter++$ ;
17         /* To move the pointer to the next element */
18         $indexStack \leftarrow$indexStack$\rightarrow$next;
19     end
20 end
21 if ($meter$ != 0) then
22     $average \leftarrow valueSim/meter$;
23 end
24 return $average$ ;
```

The variable $meter$ allows to count the number of similarity values evaluated and $valueSim$ is the sum of similarity values. These operations are repeated until there is no concept in the stack $stackConcept$. In output, the algorithm computes the average and returns it as the final result of the algorithm.
The algorithm 3 gives the average of semantic similarity values of a set of concepts in an ontology. Since edge-based similarity measures are symmetric, then instead to select $Sim(c_i, c_j, O)$ and $Sim(c_j, c_i, O)$ in the calcul, we choose one of those values, because $Sim(c_i, c_j, O) = Sim(c_j, c_i, O)$. The algorithm also does not assess the similarity between a concept and itself.

Algorithm 4 defines a method which extends an ontology O_2 by adding concepts of another ontology O_1.

```
Algorithm 4: extensionOntology. /* extension of $O_2$ by adding concepts of $O_1$ */
   input  : $stackCommon$: stack of concpts ; $O_1$, $O_2$ : Ontologies;
   output: void method which extends ontology $O_2$
1  variables : $c_1, c_2$ : concepts ;
     $stackCommonCopy, stackOfSons$ : stacks of concepts;
2  /* copy elements of stackCommon into stackCommonCopy */
3  $copyStack(stackCommon, stackCommonCopy)$ ;
4  while ($!stackCommonCopy$) do
5      $c_1 \leftarrow$depilate(stackCommonCopy);
6      /* stack all sons of $c_1$ in a stack called $stackOfSons$ ($c_1 \in O_1$) */
7      $stackOfSons \leftarrow$stackSons($c_1, O_1$) ;
8      while ($!stackOfSons$) do
9          $c_2 \leftarrow$depilate(stackOfSons);
10         if ($!findInOntology(c_2, O_2)$) then
11             /* insertion of $c_2$ as a son of $c_1$ in $O_2$ */
12             $insertAsSonOf(c_2, c_1, O_2)$ ;
13         end
14     end
15 end
```

In algorithm 4 takes in input the stacks of common concepts ($stackCommon$) of ontologies O_1 and O_2, and the two ontologies. Function $copyStack(stackCommon, stackCommonCopy)$ copies all concepts of $stackCommon$ in another stack ($stackCommonCopy$) for not loosing the elements of stack when we depile them. Once $stackCommon$ has been copied, we use its copy for searching concepts of O_1 to insert to O_2 for its extention. For each concept c_1 of O_1 stored in $stackCommonCopy$ ($c_1 \in O_1$ and $c_1 \in stackCommonCopy$), we stack all its direct sons in $stackOfSons$ thanks to function $stackSons(c_1, O_1)$. Then, for each c_2 son of c_1 ($c_2 \in O_1$), we use function $findInOntology(c_2, O_2)$ for checking if c_2 is not in O_2. If c_2 doesn't belong to O_2 ($c_2 \notin O_2$), we insert c_2 as a son of c_1 in O_2 thanks to function $insertAsSonOf(c_2, c_1, O_2)$. In result, we obtain the ontology O_2 extended by concepts of O_1.

Finally, the last algorithm (algorithm 5) implements the expression 4 defined in the section 3.2.

Algorithm 5 allows to assess similarity value between two ontologies O_1 and O_2. In input, we have the two ontologies. The ontologies are stored on stacks $stackO_1$ and $stackO_2$ thanks to a function $stack(O)$ which stores all concepts of an ontology O in a stack. After storing the concepts in the stacks, the sets of

```
Algorithm 5: simOnto. /* Assess similarity between two ontologies */
   input  : $O_1$, $O_2$ : ontology ;
   output: $valueSim$ : real
1  variables : $stackO_1$, $stackO_2$, $stackCommon$,
      $stackDiff_{(O_1 \backslash O_2)}$, $stackDiff_{(O_2 \backslash O_1)}$,
      $stackFinalO_1$, $stackFinalO_2$,
      $stackCommonFinal$ : stacks of concepts ;
2  $\alpha, \beta, \theta, \omega$ : real ;
3  $x, y, z$ : real ;
4  $n_1, n_2, I_1, I_2$ : real ;
5  $valueSim$ : real ;
6  /* Store $O_1$'s concepts in $stackO_1$ (respectively $O_2$'s
   concepts in $stackO_2$) */
7  $stackO_1 \leftarrow$stack(O$_1$) ;
8  $stackO_2 \leftarrow$stack(O$_2$) ;
9  /* create the stack of difference between $O_1$ and $O_2$
   (respectively $O_2$ and $O_1$) */
10 $stackDiff_{(O_1 \backslash O_2)} \leftarrow$diffOnto(stackO$_1$, $stackO_2$)
   ;
11 $stackDiff_{(O_2 \backslash O_1)} \leftarrow$diffOnto(stackO$_2$, $stackO_1$)
   ;
12 /* create the stack of resemblance between $O_1$ and
   $O_2$ */
13 $stackCommon \leftarrow$cCOnto(stackO$_1$, $stackO_2$) ;
14 /* extension of ontologies by adding concepts
   subsumed by common concept */
15 $extensionOntology(stackCommon, O_2, O_1)$ ;
16 $extensionOntology(stackCommon, O_1, O_2)$ ;
17 /* store concepts of extended ontologies in stacks */
18 $stackFinalO_1 \leftarrow$stack(O$_1$) ;
19 $stackFinalO_2 \leftarrow$stack(O$_2$) ;
20 /* create the stack of resemblance between $O_1$ and
   $O_2$ after their extensions */
21 $stackCommonFinal$
      $\leftarrow$cCOnto(stackFinalO$_1$, $stackFinalO_2$) ;
22 /* number of concepts of $O_1$ (respectively $O_2$) we
   need for extension of the set of common concepts */
23 $n_1$
      $\leftarrow$sizeOf(stackFinalO$_2$) $- sizeOf(stack(O_2))$ ;
24 $n_2$
      $\leftarrow$sizeOf(stackFinalO$_1$) $- sizeOf(stack(O_1))$ ;
25 /* integrity index of $O_1$ and $O_2$ */
26 $I_1 \leftarrow 1 / (1 + n_2)$ ;
27 $I_2 \leftarrow 1 / (1 + n_1)$ ;
28 /* initialization of $\alpha$, $\beta$, $\theta$ and $\omega$ */
29 $\theta \leftarrow$(sizeOf(stackCommonFinal)) /
      (sizeOf(stackO$_1$) $+ n_1 + n_2$) ;
30 $\omega \leftarrow$(sizeOf(stackCommonFinal)) /
      (sizeOf(stackO$_2$) $+ n_1 + n_2$) ;
31 $\alpha \leftarrow$sizeOf(stackDiff$_{(O_1 \backslash O_2)}$)$/sizeOf(stackO_1)$ ;
32 $\beta \leftarrow$sizeOf(stackDiff$_{(O_2 \backslash O_1)}$)$/sizeOf(stackO_2)$ ;
33 /* partial computation of similarity values */
34 $x \leftarrow (\theta \times aSV(stackCommonFinal, O_1) + I_2)+$
      $(\omega \times aSV(stackCommonFinal, O_2) + I_1)$ ;
35 $y \leftarrow \alpha \times aSV(stackDiff_{(O_1 \backslash O_2)}, O_1)$ ;
36 $z \leftarrow \beta \times aSV(stackDiff_{(O_2 \backslash O_1)}, O_2)$ ;
37 /* Computation of similarity values between $O_1$ and
   $O_2$ */
38 if $((x+y+z) != 0)$ then
39 |  $valueSim \leftarrow$x/(x+y+z);
40 end
41 else
42 |  $valueSim \leftarrow -1$;
43 end
44 return $valueSim$ ;
```

resemblance ($stackCommon$) and difference ($stackDiff_{(O_1 \backslash O_1)}, O_2$) and $stackDiff_{(O_1 \backslash O_2)}, O_1$)) are determined by calling the algorithms 1 and 2. Once the sets have been determined, we extend ontologies thanks to the function $extensionOntology(stackCommon, O_2, O_1)$ which extends O_1 (respectively O_2) by adding sons of concepts of O_2 (respectively O_1) included in the stack $stackCommon$ but not in O_1 (respectively O_2). After extending the ontologies, we redefine the stacks of concepts of ontologies for re-evaluating the stack of concepts that they share ($stackCommonFinal$). We also compute n_1 and n_2 respectively the number of concepts of O_1 added to O_2, and the number of concepts of O_2 added to O_1. Then, we assess the integrity indexes I_1 and I_2 before initializing parameters α, β, θ and ω. Finally, we compute similarity of two ontologies and return the final result. The result is equal to -1 if there are errors in the calculation process.

4 EXPERIMENTATIONS

Example 1:

The example 1 is about a fragment of Wordnet[1] that we have used in our previous works (Ngom et al., 2016b) and (Ngom et al., 2016a). The ontologies are represented by figures 2 and 3.

We obtain the following results:

- $aSV(stackCommonFinal, O'_3) = 0.5$;
- $aSV(stackCommonFinal, O'_4) = 0.5$;
- $aSV(stackDiff_{(O_3 \backslash O_4)}, O_3) = 0$;

[1]http ://wordnet.princeton.edu

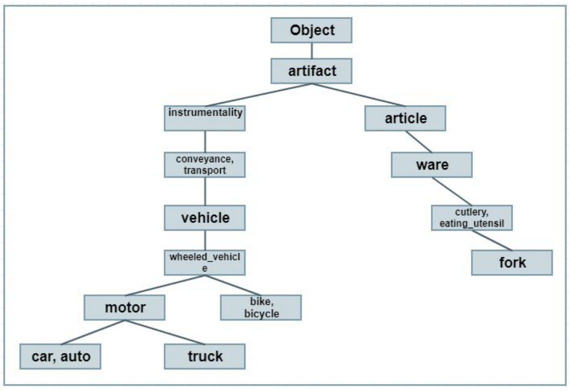

Figure 2: Representation of an ontology extracted from WordNet (O_3).

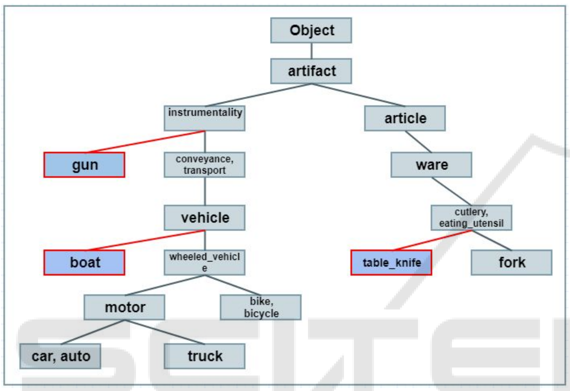

Figure 3: Representation of an ontology extracted from WordNet and extended with some concepts (O_4).

- $aSV(stackDiff_{(O_4 \backslash O_3)}, O_4) = 0.2875$;
- $\theta = 1$; $\omega = 0.85$; $\alpha = 0$; $\beta = 0.18$;
- $I_3 = 1/4$; $I_4 = 1$; $n_3 = 0$; $n_4 = 3$;
- $N_{Plus}(O_3, O_4) = 0.98$

In this example, we illustrate our proposition by assessing the similarity between the ontology of the figure 2 and that of the figure 4.

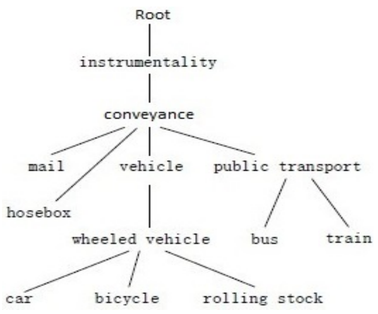

Figure 4: Representation of an ontology extracted from WordNet (O_5).

We obtain the following results:

- $aSV(stackCommonFinal, O'_3) = 0.69$;
- $aSV(stackCommonFinal, O'_5) = 0.62$;
- $aSV(stackDiff_{(O_3 \backslash O_5)}, O_3) = 0.51$;
- $aSV(stackDiff_{(O_5 \backslash O_3)}, O_5) = 0.68$;
- $\theta = 11/19$; $\omega = 11/18$; $\alpha = 3/7$; $\beta = 6/13$;
- $I_3 = 1/5$; $I_5 = 1/2$; $n_3 = 1$; $n_5 = 4$;
- $N_{Plus}(O_3, O_5) = 0.74$

5 ANALYSIS AND COMPARISONS

In the Experimentation section (section 4), we have given two examples for illustrating our proposition. This section is about analysis of obtained results. We also compare this results to the results obtained in (Ngom et al., 2017) with same examples.

Example 1:

Ontologies O_3 and O_4 present a good similarity value ($N_{Plus}(O_3, O_4) = 0.98$). Then, we can say that the similarity value between two ontologies is good. We note that ontologies have respectively 14 concepts for O_3 and 17 concepts for O_4. The ontology O_4 contains all O_3's concepts and 3 more concepts. Initially, in the different sets, we have 14 concepts in the stack *stackCommon* which represents the set $(O_3 \cap O_4)$, 0 concept in the set $(O_3 \backslash O_4)$ stored in the stack $stackDiff_{(O_3 \backslash O_4)}$ and 3 concepts in the set $(O_4 \backslash O_3)$ stored in the stack $stackDiff_{(O_4 \backslash O_3)}$. After extension of ontologies, we have 17 concepts in O'_3 and 17 concepts in 17 concepts in O'_4. O'_4 does not change but O'_3 has 3 more concepts than O_3. In O'_3, the insertions of concepts have been done following this rules : "gun" is inserted as a son of "instrumentality" ("gun" is-a "instrumentality"), "boat" as a son of "vehicle" ("boat" is-a "vehicle") and "table_knife" as a son of "cutlery, eating_utensil" ("table_knife" is-a "cutlery, eating_utensil"). Finally, the set $(O_3 \cap O_4)$ becomes $(O'_3 \cap O'_5)$ and contains 17 concepts stored in the stack *stackCommonFinal*.

Example 2:

The similarity value between ontologies O_3 and O_5 is equal to 0.74 ($N_{Plus}(O_3, O_5) = 0.57$). This similarity value is good. The ontologies have respectively 14 concepts for O_3 and 13 for O_5. Before extending ontologies, in the different sets, we have 7 concepts in the stack *stackCommon* which represents the set $(O_3 \cap O_5)$, 6 concept in the set $(O_3 \backslash O_5)$ stored in the stack $stackDiff_{(O_3 \backslash O_5)}$ and 6 concepts in the set $(O_5 \backslash O_3)$ stored in the stack $stackDiff_{(O_5 \backslash O_3)}$. After the extension of ontologies, we have respectively 18 concepts in O'_3 and 14 concepts in O'_5. The set $(O_3 \cap O_5)$ is extended and becomes $(O'_3 \cap O'_5)$. $(O_3 \cap O_5)$ contains 11 concepts stored in the stack *stackCommonFinal*. In O'_3, the insertions of concepts have been done following this rules : in first, ”mail”, ”hosebox” and ”public_transport” are inserted as a sons of ”conveyance” (”mail” is-a ”conveyance”, ”hosebox” is-a ”conveyance” and ”public_transport” is-a ”conveyance”), and then, ”rolling_stock” is inserted as a son of ”Wheeled_vehicle” (”rolling_stock” is-a ”wheeled_vehicle”). The ontology O'_5 is extended following that ”motor” is inserted as a son of ”wheeled_vehicle” (”motor” is-a ”wheeled_vehicle”).

The Table 1 allows to compare the results obtained in this paper to the results of our previous works (Ngom et al., 2017).

Table 1: Comparison of results of N_{Plus} and T_{Ngom}.

	N_{Plus}	T_{Ngom}
(O_3, O_4)	0.98	0.95
(O_3, O_5)	0.74	0.57

In Table 1, we find that the measure N_{Plus} gives better results compared to the measure T_{Ngom}. Indeed, we have : $N_{Plus}(O_3, O_4) > T_{Ngom}(O_3, O_4)$ (0.98 > 0.95) and $N_{Plus}(O_3, O_5) > T_{Ngom}(O_3, O_5)$ (0.74 > 0.57). The N_{Plus} measure increases the T_{Ngom} measure thanks to the extension of the set of their common concepts.

6 CONCLUSION

In this paper, we proposed an approach for assessing similarity between two ontologies. The approach that we adopt is based on set theory, edges based semantic similarity (Ngom., 2015) and feature-based similarity (Tversky, 1977). It can be summarized in 5 steps. In **step 1**, we have determined the sets of concepts which characterize the concepts shared by the two ontologies and the sets of concepts that are different from them. Once the sets have been determined, we have assessed the average of the semantic similarity values between concepts of each set in **step 2**. **Step 3** is dedicated to the extension of ontologies. After the step 3, we have extended set of concepts that ontologies share in **Step 4**. Finally, we assessed similarity between ontologies in **step 5**. The approach proposed in this paper improve our proposition in (Ngom et al., 2017) by extending the set of concepts shared by the two ontologies. The extension on this set is realized by taking into account the concepts of ontologies linked with concepts of the set by the ”is-a” relation. In perspectives, we will propose an approach to assess similarity between an ontology and a speech in text format to check if the text and the ontology refer to the same theme.

ACKNOWLEDGMENT

The authors would like to thank the CEA-MITIC (The African Center of Excellence in Mathematics, Computer Science and ICT) which partially funded this work.

REFERENCES

Dramé, K. (2014). *Contribution à la construction d’ontologies et à la recherche d’information : application au domaine médical.* PhD thesis, Université De Bordeaux.

Levenshtein., I. V. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Cybernetics and Control Theory, 10(8)*, pages 707 – 710.

Maedche, A. and Staab, S. (2002). Measuring similarity between ontologies. In *In: Gómez-Pérez A., Benjamins V.R. (eds) Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web. EKAW 2002. Lecture Notes in Computer Science, vol 2473. Springer, Berlin, Heidelberg*, pages 251 – 263.

Ngom., A. N. (2015). Etude des mesures de similarité sémantique basée sur les arcs. In *CORIA, Paris, France*, pages 535 – 544.

Ngom, A. N., Diallo, P. F., Kamara-Sangaré, F., and LO, M. (2016a). A method to validate the insertion of a new concept in an ontology. In *SITIS 2016 : The 12th International Conference on Signal Image Technology and Internet Systems, Naples*, pages 275 – 281.

Ngom, A. N., Kamara-Sangaré, F., and Lo, M. (2017). Proposition of a method for assessing similarity between two ontologies. In *4th Annual Conf. on Computational Science & Computational Intelligence (CSCI’17) — Dec 14-16, 2017 — Las Vegas, Nevada, USA*, pages 174 – 179.

Ngom, A. N., Traore, Y., Diallo, P., Sangare, F., and Lo, M. (2016b). A method to update an ontology : simulation. In *Int’l Conf, Information and Knowledge Engineering, IKE’16, Las Vega, Nevada, USA*, pages 92 – 96.

Tversky, A. (1977). Features of similarity. In *Psychological Review, 84(4)*, pages 327– 352.

Wu, Z. and Palmer, M. (1994). Verbs semantics and lexical selection. In *U.A.f.C.L. Stroudsburg, PA (ed.), In Proceedings of the 32nd annual meeting on ACL, volume 2 de ACL ’94*, pages 133 – 138.